

Programação III

Jordana S. Salamon

jssalamon@inf.ufes.br

jordanasalamon@gmail.com

DEPARTAMENTO DE INFORMÁTICA

CENTRO TECNOLÓGICO

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

Orientação a Objetos - Um novo paradigma

Programação Estruturada x Programação Orientada a Objetos

▶ Programação Estruturada

- ▶ + Controle mais eficaz quanto ao fluxo de execução do programa
- ▶ + Facilidade de compreender o código quando o mesmo é analisado
- ▶ + Desempenho do código maior que linguagens OO
- ▶ - Código confuso, caso o desenvolvedor não faça a modularização do código
- ▶ - Dificuldade em realizar o reuso
- ▶ Obs: Adequada para construir programas para realização de operações lógicas e aritméticas.



Programação Estruturada x Programação Orientada a Objetos

▶ Programação OO

- ▶ + Reutilização de código
- ▶ + Melhor organização do código no programa
- ▶ + Mais próxima do Ser Humano
- ▶ - Desempenho do código menor que linguagens estruturadas
- ▶ - Confusão do código caso os conhecimentos de OO não sejam corretamente aplicados
- ▶ Obs: Adequada para construir Sistemas de Informação.

Orientação a objetos não é mágica e nem a “tábua de salvação” do desenvolvimento. É preciso aplicá-la com disciplina e em conjunto com outras técnicas da Engenharia de Software.

Linguagens do Paradigma OO



Linguagens do Paradigma OO

main.java

```
class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello world!");  
    }  
}
```

main.cpp

```
#include <iostream>  
  
int main() {  
    std::cout << "Hello World!\n";  
}
```

main.cs

```
using System;  
  
class MainClass {  
    public static void Main (string[] args) {  
        Console.WriteLine ("Hello World");  
    }  
}
```

main.py

```
def main():  
    print("Hello World")  
  
if __name__ == '__main__':  
    main()
```

main.go

```
package main  
  
import "fmt"  
  
func main() {  
    fmt.Println("Hello World")  
}
```

main.kt

```
fun main(args: Array<String>) {  
    println("Hello, world!")  
}
```



Linguagens do Paradigma OO

main.java

```
class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello world!");  
    }  
}
```

main.cpp

```
#include <iostream>  
  
int main() {  
    std::cout << "Hello World!\n";  
}
```

main.cs

```
using System;  
  
class MainClass {  
    public static void Main (string[] args) {  
        Console.WriteLine ("Hello World");  
    }  
}
```

main.py

```
def main():  
    print("Hello World")  
  
if __name__ == '__main__':  
    main()
```

main.go

```
package main  
  
import "fmt"  
  
func main() {  
    fmt.Println("Hello World")  
}
```

main.kt

```
fun main(args: Array<String>) {  
    println("Hello, world!")  
}
```



Java x C++

- ▶ Java é uma linguagem de programação de uso geral extremamente popular, projetada para funcionar em praticamente qualquer dispositivo. É a linguagem de escolha para aplicativos da Web cliente-servidor, com 9 milhões de desenvolvedores usando a plataforma para essa finalidade.
- ▶ Quando os programadores pensam em Java, eles geralmente pensam nas seguintes coisas:
 - ▶ Servidores de aplicativos
 - ▶ Aplicativos da web
 - ▶ Aplicações móveis
 - ▶ Aplicativos de desktop e corporativos
- ▶ Java também é uma das linguagens de escolha para testes de unidade. Os programadores Java podem aprender a fazer isso para aplicativos robóticos e de Internet das Coisas (IoT).
- ▶ IoT é uma área importante do desenvolvimento Java.
- ▶ C ++ é uma linguagem de programação de baixo nível comumente usada para grandes projetos de infraestrutura de software e para projetos de software embarcados.
- ▶ C ++ é a linguagem escolhida para algumas aplicações muito importantes:
 - ▶ Aplicações institucionais de grande porte, como as usadas por bancos, governos e outras instituições.
 - ▶ Software embarcado projetado para operar robôs, satélites, eletrônicos de consumo e outros dispositivos de hardware.
 - ▶ Jogos com gráficos pesados e aplicações científicas.
- ▶ Enquanto o Java é uma linguagem fácil de aprender e escrever, o que reduz o tempo de desenvolvimento, o C ++ produz o código mais enxuto e eficaz para aplicações de alto impacto.

Java x C++

- ▶ Java também é uma linguagem importante para jogos móveis e baseados em navegador.
- ▶ O Android depende do Java para uma ampla gama de jogos e de alguns dos seus aplicativos mais populares.
- ▶ Programas escritos em C ++ tendem a usar os recursos do computador de forma mais eficiente do que aqueles escritos em Java ou outras linguagens.
- ▶ Mas se você quer desenvolver jogos para plataformas de console e PC, então C ++ é o melhor lugar para começar.

Se você quiser construir um web scraper capaz de manipular muitos dados, aprenda Python ou Java.

Se você deseja criar aplicativos para dispositivos móveis, concentre-se no Java ou no Swift da Apple.

Se você curte videogames baseados em PC e console, comece com C ++.

Se você quiser analisar muitos dados ou escrever programas de aprendizado de máquina, aprenda Python ou R.

Se escrever sistemas embarcados para fazer funções de hardware fascinar você, vá em direção ao C ++.

Se você quiser entrar no mundo do desenvolvimento de IoT, confie em Java ou C ++ para levá-lo até lá.

Fonte: <https://javabeat.net/java-vs-c/>



nemo

Introdução Orientação a Objetos

Modelagem Conceitual Estrutural

O Paradigma Orientado a Objetos

- O mundo é visto como sendo composto por objetos, onde um objeto é uma entidade que combina estrutura de dados e comportamento funcional.



Carro

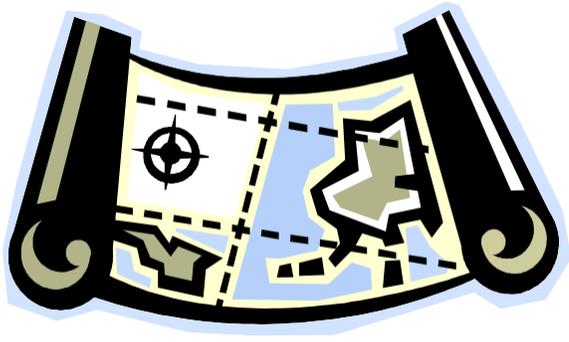


Pessoa

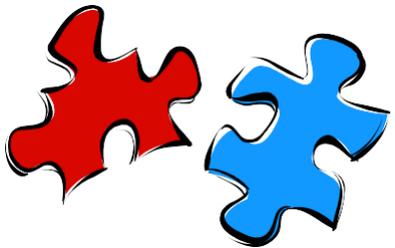


Turma

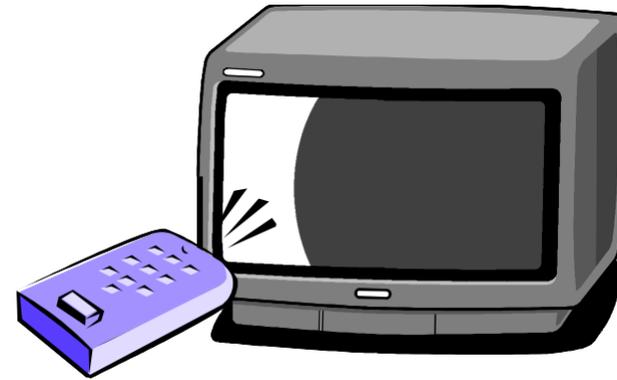
Princípios OO



Abstração



Modularização



Encapsulamento



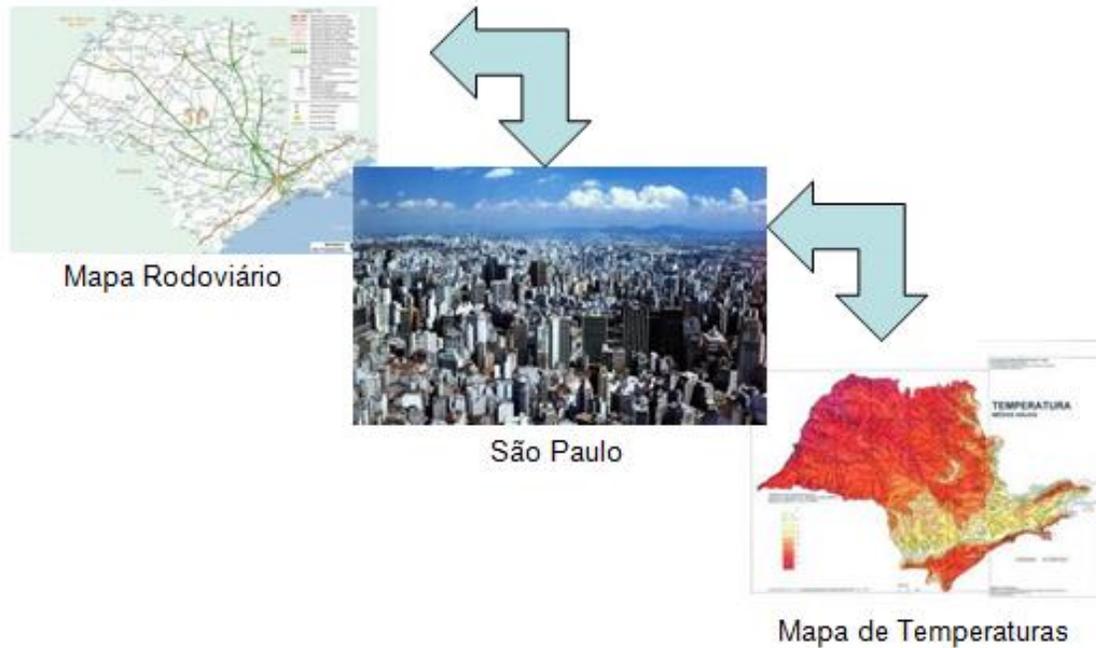
Herança

Princípios OO

Princípios da Orientação a Objetos

a) Abstração

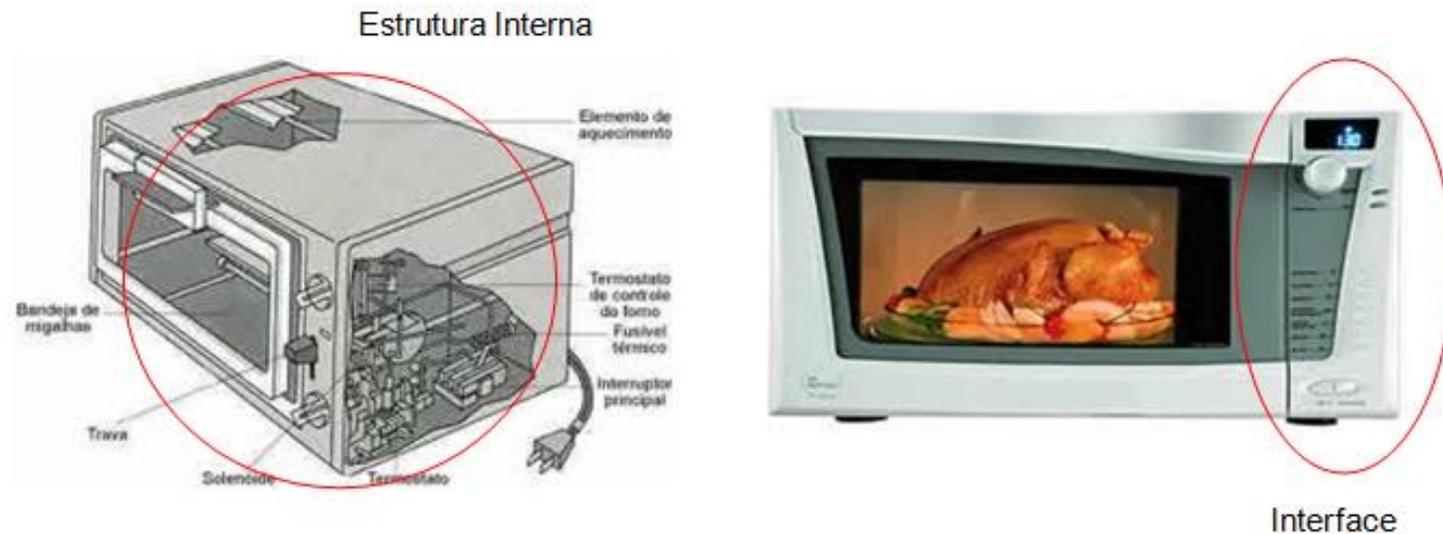
Visão simplificada de algo, onde apenas os elementos relevantes são considerados.



Princípios OO

b) Encapsulamento

- Separação dos aspectos externos de um objeto, acessíveis por outros objetos, de seus detalhes internos de implementação, que ficam ocultos dos demais objetos.

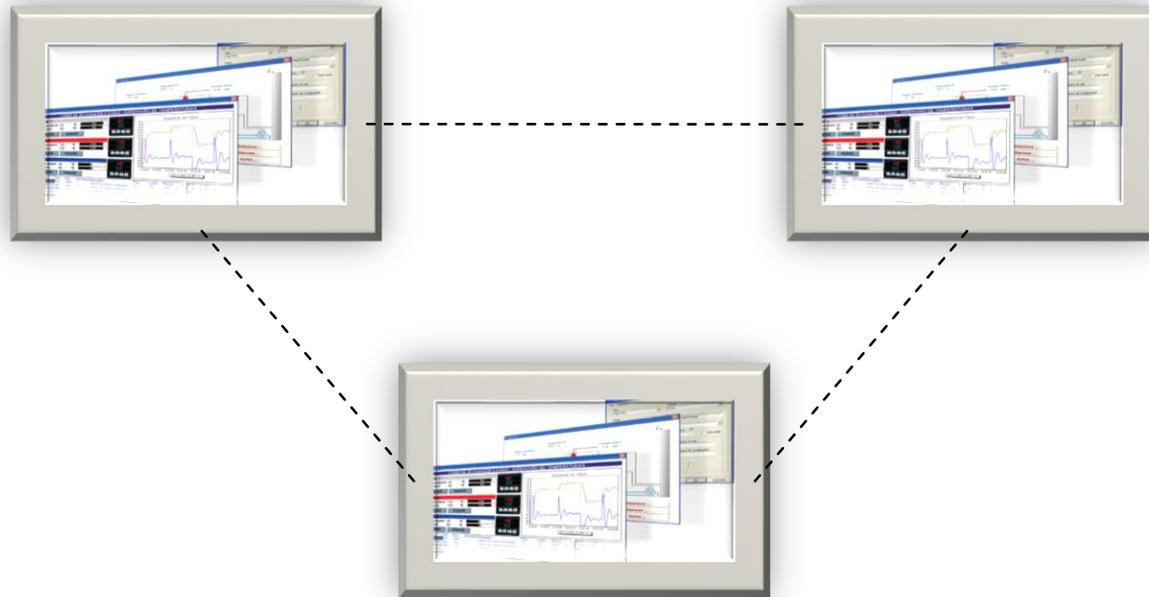


- Abstração e encapsulamento são conceitos complementares: enquanto a abstração enfoca o comportamento observável de um objeto, o encapsulamento oculta a implementação que origina esse comportamento.

Princípios OO

c) Modularidade

Obtenção de sistemas decompostos em um conjunto de módulos coesos e fracamente acoplados.



Princípios OO

Conceitos da Orientação a Objetos

- a) **Objetos:** entidades que interagem entre si, onde cada uma delas desempenha um papel específico.



O carro do
João



João

- b) **Classes:** descrevem um conjunto de objetos com as mesmas propriedades (atributos e associações) e o mesmo comportamento (operações).

Objetos são instâncias das classes



Carro A

Carro



Carro B



Carro C



João

Pessoa



Maria



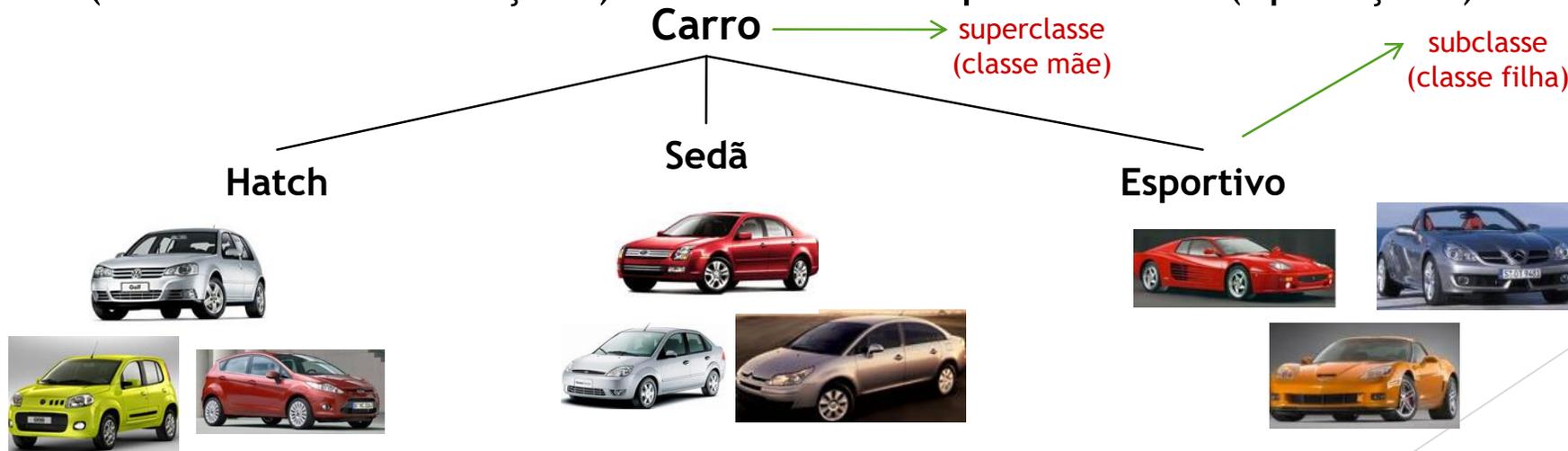
Cecy

Princípios OO

c) **Ligações e Associações:** relacionamentos entre objetos e classes (respectivamente).

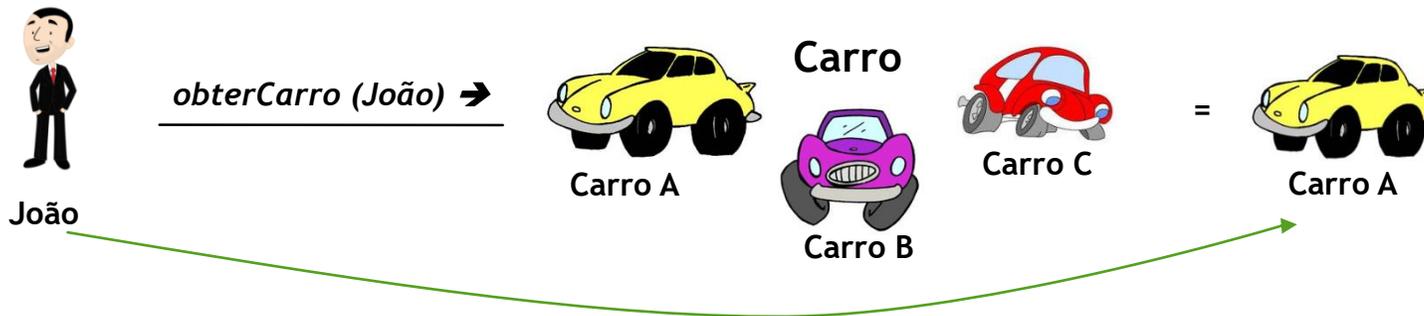


d) **Herança:** descrevem um conjunto de classes com as mesmas propriedades (atributos e associações) e o mesmo comportamento (operações).

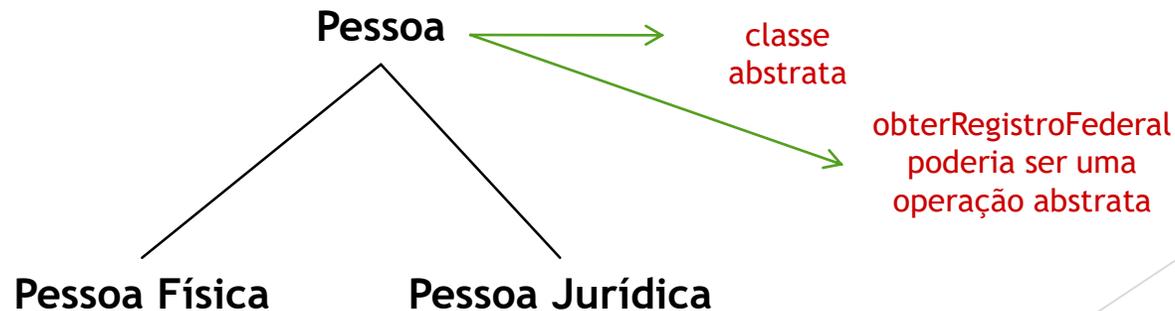


Princípios OO

e) **Mensagens e Métodos:** forma de comunicação entre os objetos.

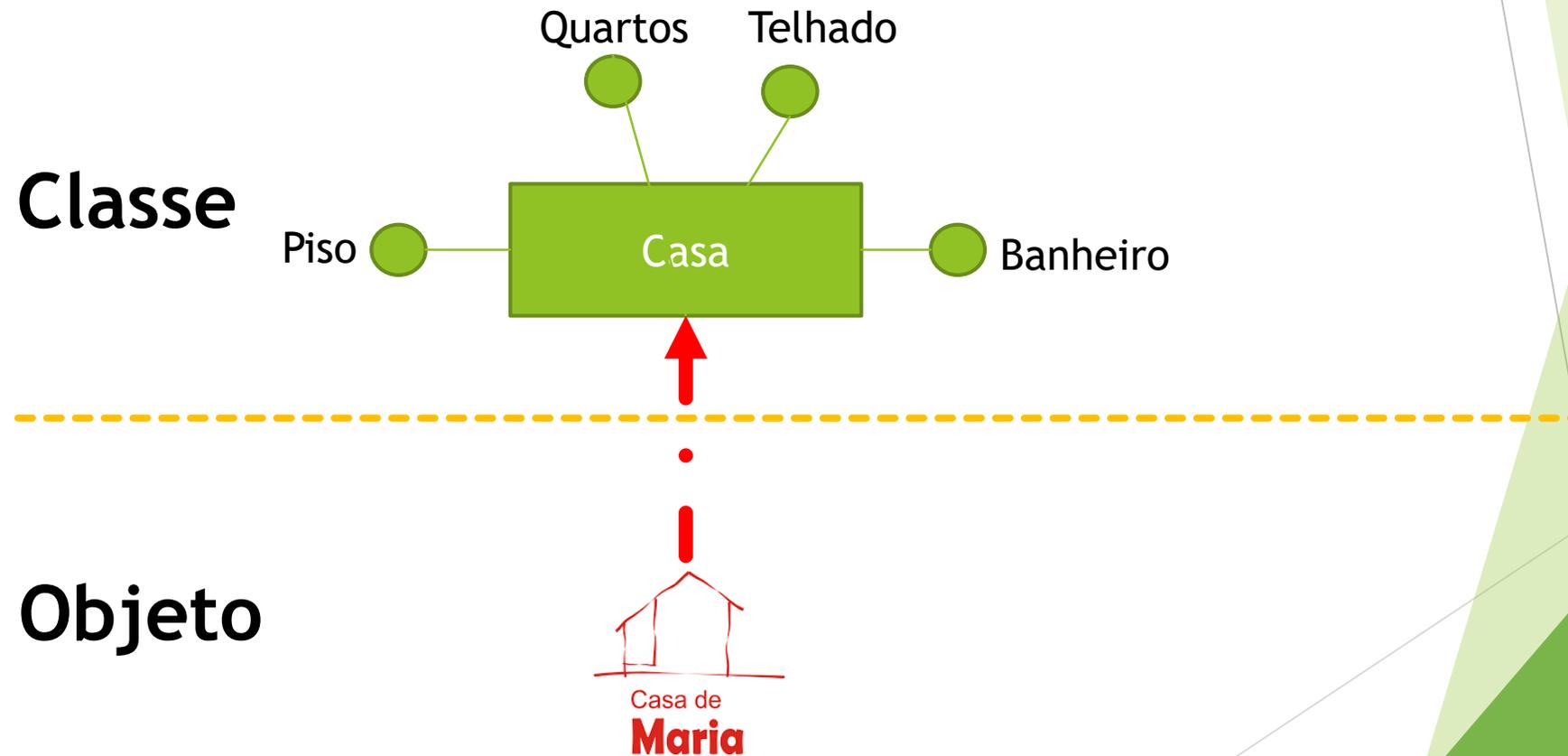


d) **Classes e Operações Abstratas:** classes que não possuem instância e operações que não são implementadas nas classes (são apenas assinatura). A classe abstrata existe meramente para que um comportamento comum a um conjunto de classes possa ser colocado em uma localização comum e definido uma única vez.



O que é um Objeto?

- ▶ Um Objeto é instância de uma **Classe**



Classe, Atributo ou Objeto?

Livro

Livro de Programação

Fusca

Torre Eiffel

Sol

Endereço

Telefone



nemo

Classe, Atributo ou Objeto?

Livro

Livro de Programação

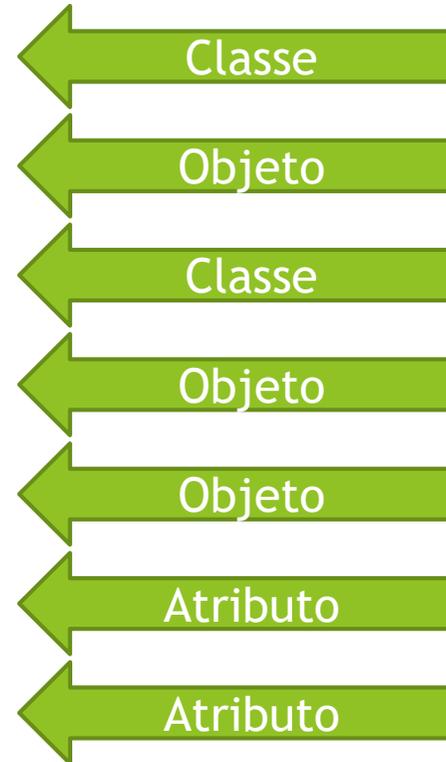
Fusca

Torre Eiffel

Sol

Endereço

Telefone



O que é uma Classe?

- ▶ Semelhante a uma estrutura de C (struct).
- ▶ Define os atributos (variáveis) e métodos (funções) para um determinado Tipo de Objeto (por exemplo Aluno).
- ▶ Encapsula operações relativas a esse tipo de Objeto;



Exemplo

Classe Estudante



Nome

Idade

Atributos

Métodos:

```
void chorarNota(){
    while(1){
        printf("ponto extra");
    }
}

void pedirProvaEmDupla(){
    while(1){
        printf("Professor, podemos fazer a prova em Dupla?");
    }
}

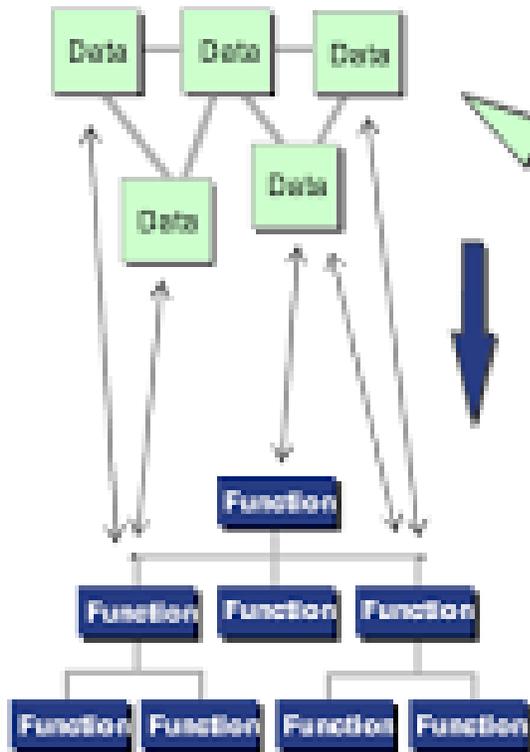
void pedirAdiamentoDoTrabalho(){
    while(1){
        printf("Professor, adia o trabalho, não consegui terminar")
    }
}
```



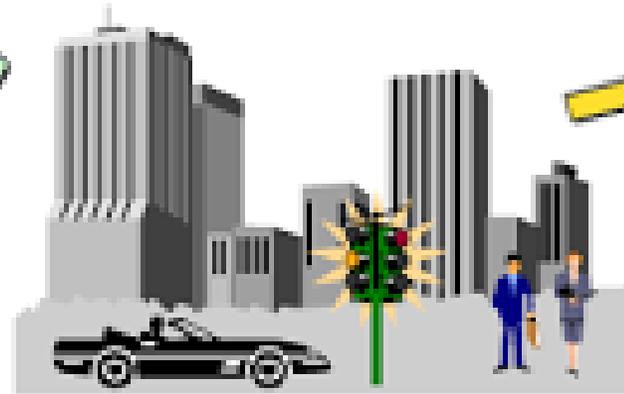
Comparação Procedural x POU

Procedural:

Separation of data and functions

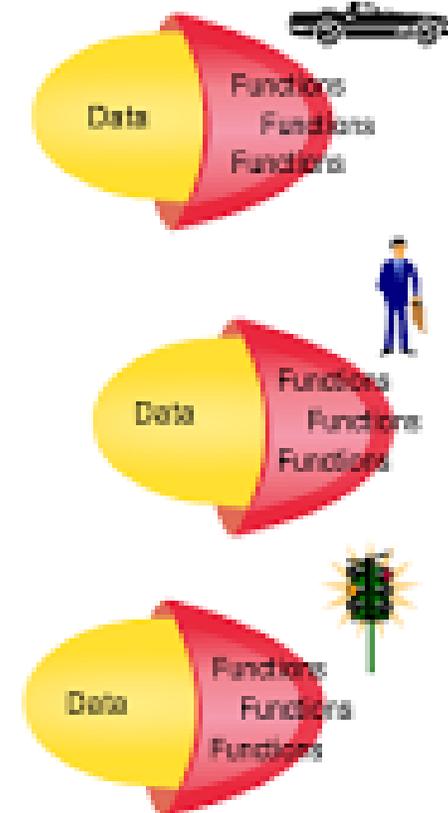


Real world



Object-oriented:

Encapsulation of data and functions



Comparação Procedural x POO

Estruturado



efetuarEmprestimoLivro()

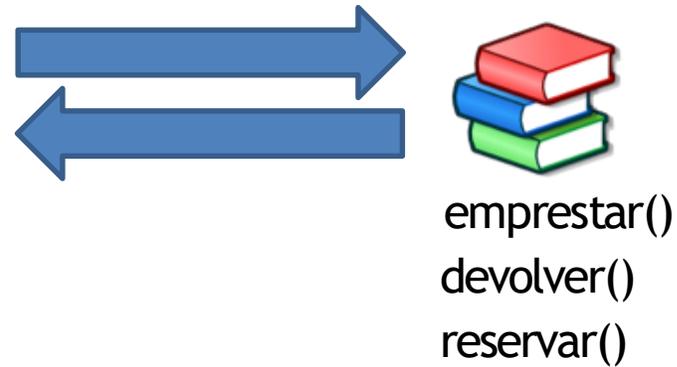


efetuarDevolucaoLivro()



efetuarReservaLivro()

Orientado a Objetos



Diferentes visões de mundo!!!

Linguagem Procedural

```
#include <stdio.h>

struct retangulo{
    float base;
    float altura;
    float area;
};

typedef struct retangulo Retangulo;

float calculaAreaRetangulo(Retangulo r) {
    return r.base * r.altura;
}

int main() {
    Retangulo r;
    r.base = 10;
    r.altura = 10;
    r.area = calculaAreaRetangulo(r);
    printf("Area = %f\n", r.area);

    return 0;
}
```

Linguagem OO

```
Classe Retangulo{  
    float altura;  
    float base;  
    float area;  
  
    void calculaArea(){  
        area = altura*base;  
    }  
}
```

Retangulo ret1;

ret1 = novo Retangulo();

ret1.altura = 10;

ret1.base = 10;
ret1.calculaArea();

escreva(ret1.area);

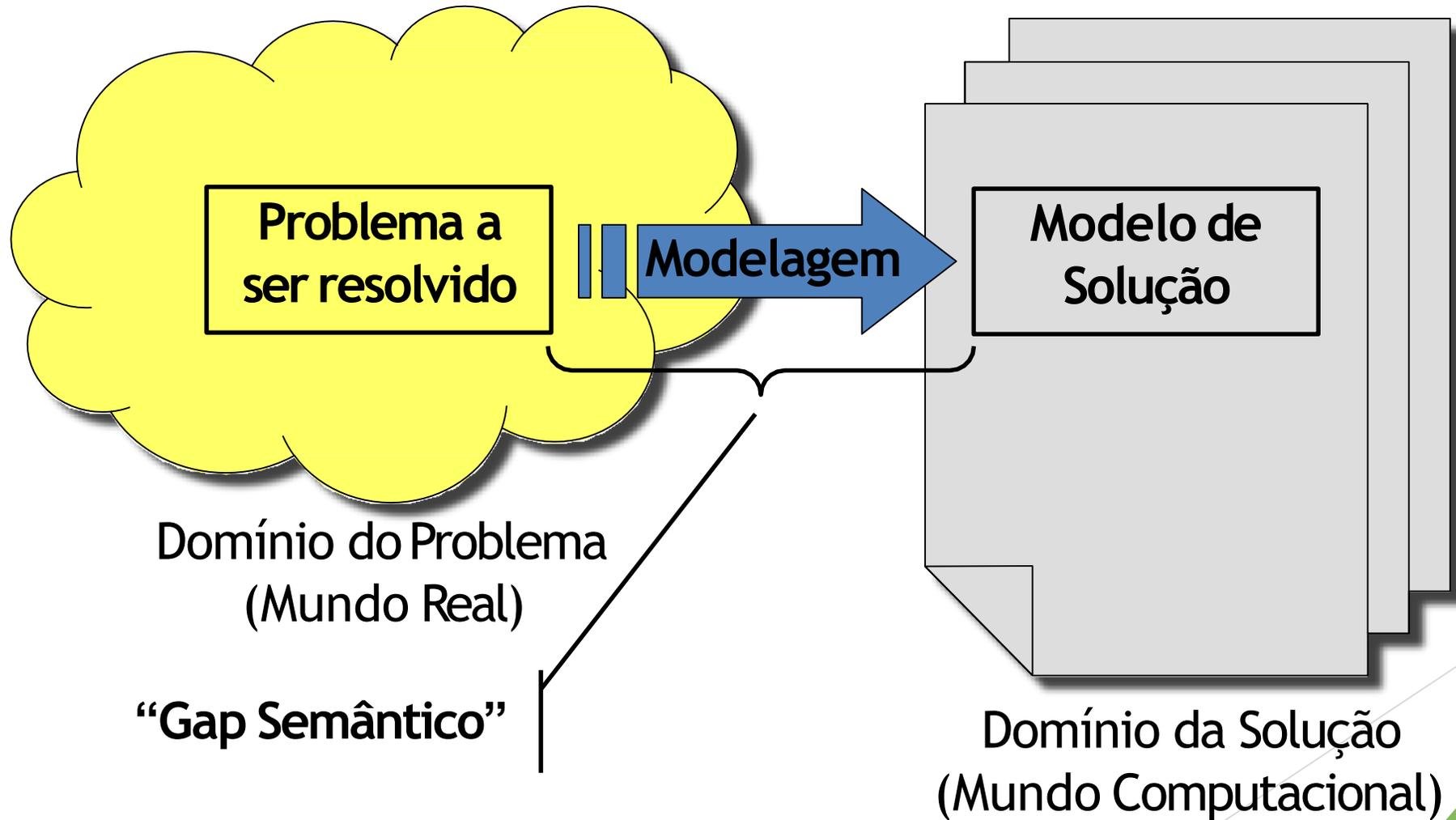


Modelagem e Projeto Orientado a Objetos

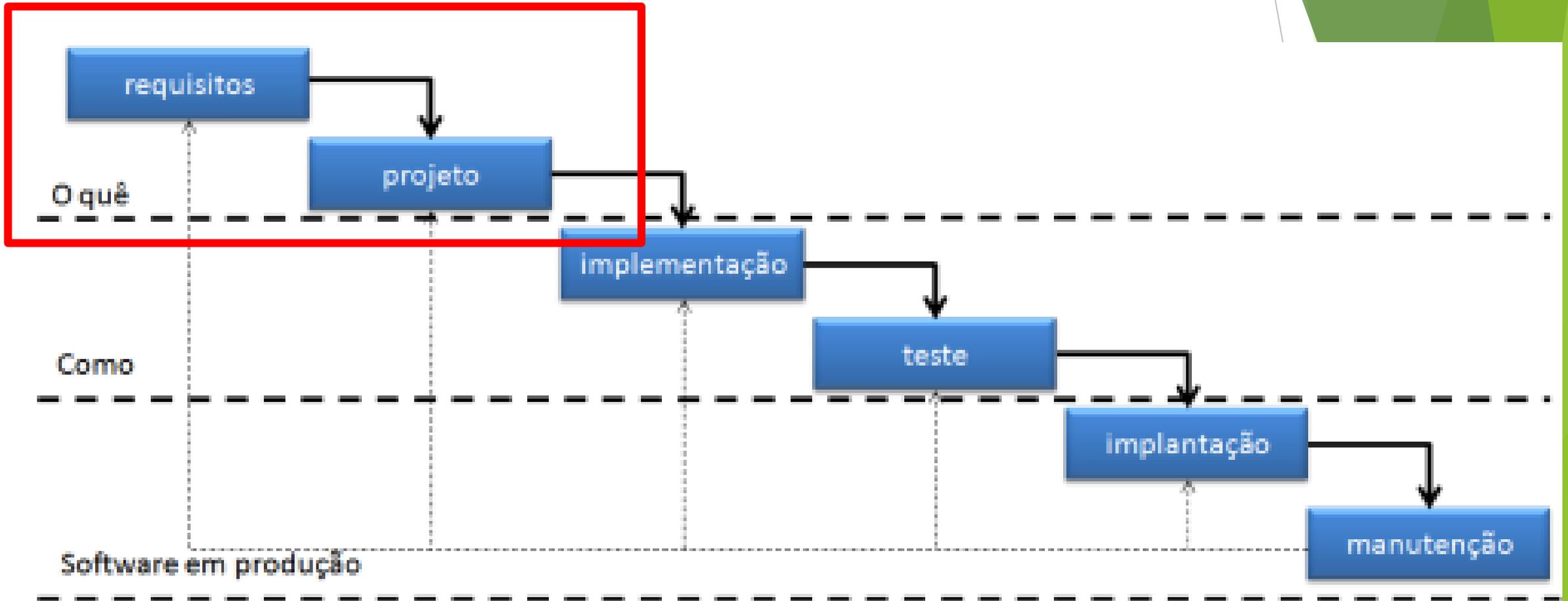


nemo

Gap semântico



Etapas para construção de um software

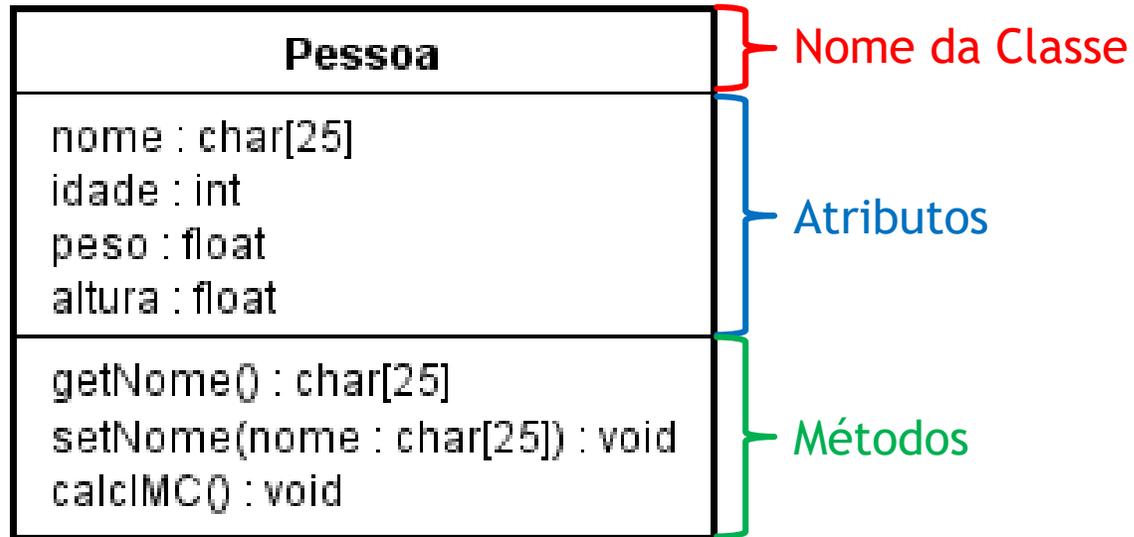


UML (Unified Modeling Language)

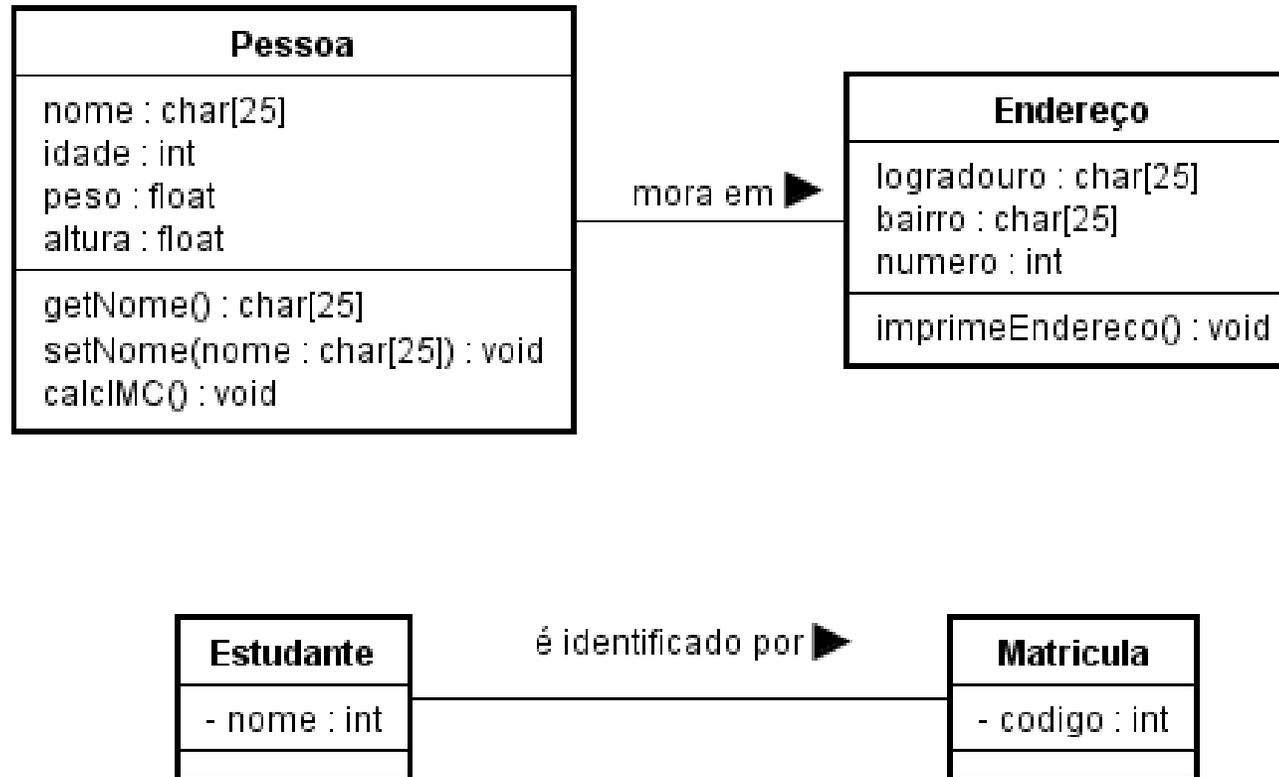
- ▶ Padrão “de facto” para especificar, visualizar, documentar e construir artefatos de um sistema desenvolvido sob o paradigma Orientado a Objetos;
- ▶ Amplamente Utilizada
- ▶ Sintaxe Visual Simples



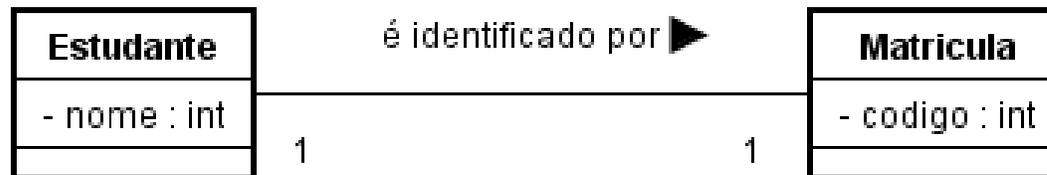
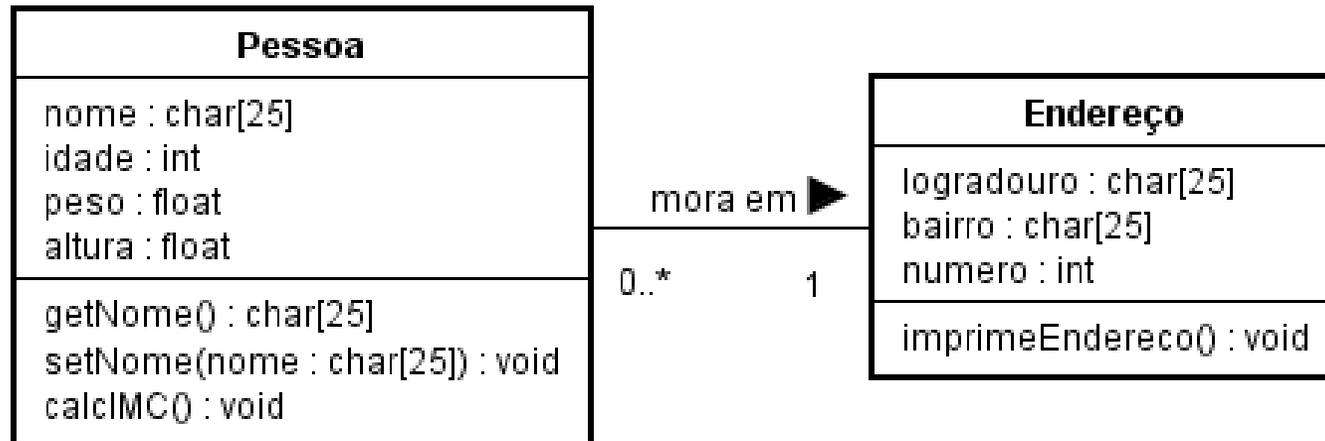
Exemplo Classe, Atributo, Método



Exemplo Relacionamento



Exemplo Cardinalidade

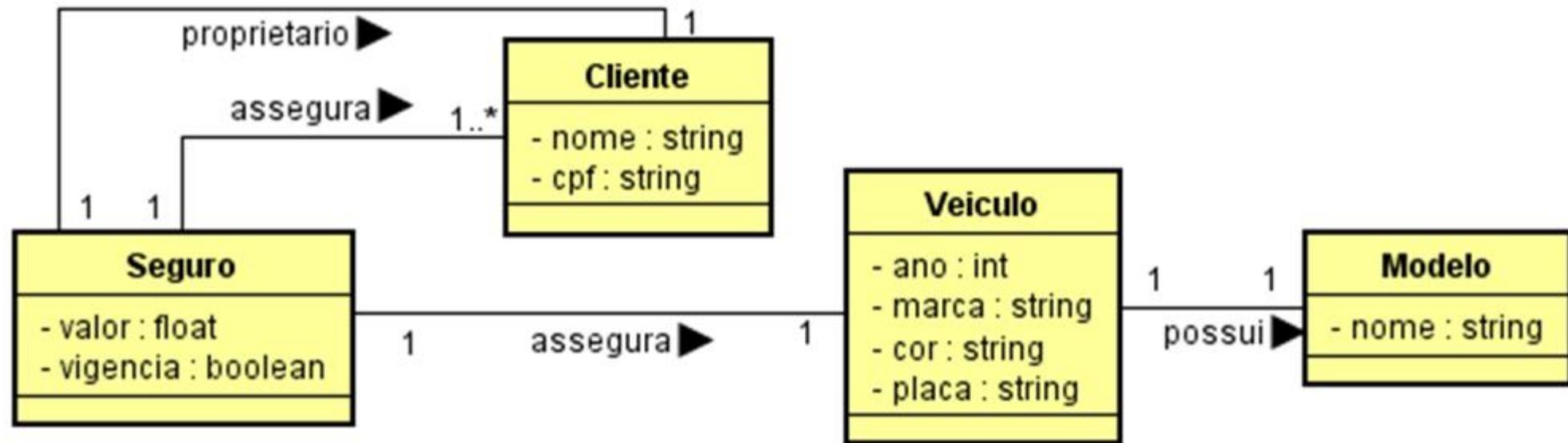


Exercício

- ▶ Desenvolva um domínio de seguros de carro, modelando as seguintes classes:
 - ▶ Carro: ano, marca, modelo, cor, placa;
 - ▶ Modelo (Carro): nome;
 - ▶ Seguro: carro, cliente, valor, vigência;
 - ▶ Cliente: nome, cpf;



Resolução



That's all Folks!



nemo